

1er Parcial

Algoritmos y Estructuras de Datos 3 – DC, FCEyN, UBA

14/05/2021

Para realizar consultas, deben conectarse por Discord al canal de le docente a le cual quieran consultar. Tener en cuenta que un docente no puede conectarse con dos estudiantes en simultáneo. Las aclaraciones de enunciado que podamos llegar a hacer van a ser comunicadas vía Discord al canal de consultas de la práctica.

El examen transcurre de 17:00 a 21:00 hs. A las 21:00 se desconectarán los docentes de sus canales de Discord y tendrán hasta las 21:30 para realizar la entrega vía campus. El archivo subido al campus puede sobreescribirse una cantidad ilimitada de veces hasta la hora de entrega. Independientemente de si sobreescriben o no, deberán confirmar su entrega definitiva (que ya no podrá sobreescribirse). Sólo en caso de que el Campus estuviera saturado y no funcionara, sería adecuado realizar la entrega por mail a algo3-doc@dc.uba.ar con copia a fsoulin@dc.uba.ar indicando claramente la entrega en el asunto.

El examen debe **realizarse a mano**. Deben **numerar sus hojas** y escribir en ellas sus **nombres y número de legajo** (i.e., libreta o DNI). Al finalizar, deben **escanearlo o fotografiarlo** y deben **unir y comprimir** las páginas resultantes para generar un único archivo en **formato PDF** con un **peso razonable**. El resultado debe ser un documento **legible** (buena iluminación, buena resolución, buena orientación, no fotos cortadas, etc.), **¡verificarlo!**. El archivo debe estar nombrado **apellido_nombre.extensión** y debe haber un **orden de lectura claro**.

El examen es personal y pueden usar las teóricas, las clases prácticas y las guías de ejercicios, citando claramente. Las respuestas deben estar debidamente justificadas incluso en aquellos ejercicios en los que este hecho no es recordado.

El examen se **aprueba** con al menos 2 ejercicios bien.

- 1) Carle quiere convertirse en le mejor speedrunner de GASTAVIDAS. GASTAVIDAS es un juego con n niveles que deben ser pasados uno después del otro. En GASTAVIDAS, le jugador empieza con una sola *vida* y en cada nivel i puede conseguir hasta v_i vidas, para algún $0 \leq v_i \leq n$. La ventaja de conseguir muchas vidas es que las mismas pueden gastarse en los distintos niveles para acelerar la resolución del juego. Por ejemplo, si un nivel tiene un piso de espinas, le jugador puede pasar por encima de ellas en unos pocos segundos perdiendo una vida, o puede buscar un camino alternativo y más largo sin gastar vidas. Obviamente, las vidas solo se pueden gastar luego de ser conseguidas y el juego termina una vez que le jugador se queda sin vidas. Para ser considerado le mejor, Carle debe participar en la categoría *World Class* que tiene tres reglas particulares: 1. cada jugador debe recolectar todas las vidas de todos los niveles, 2. las vidas recolectadas en el nivel i no se

pueden gastar en el nivel i y, 3. en cada nivel se pueden gastar a lo sumo 5 vidas. Luego de un arduo entrenamiento, Carle ha logrado resolver cada nivel i en tiempo $t_{ip} \in \mathbb{N}$ cuando gasta p vidas, para $0 \leq p \leq 5$. Ciertamente, $t_{ip} \geq t_{ip+p'}$ para todo $0 \leq p \leq p+p' \leq 5$.

- a) Definir en forma recursiva la función $G: \{1, \dots, n\} \times \mathbb{N} \rightarrow \mathbb{N}$ tal que $G(i, v)$ denota la mínima cantidad de tiempo que le toma a Carle resolver los niveles i, \dots, n si empieza con v vidas.
 - b) Demostrar que G tiene la propiedad de superposición de subproblemas.
 - c) Definir un algoritmo *top-down* para calcular $G(i, v)$ cuando $V = \{v_i \mid 1 \leq i \leq n\}$ y $T = \{t_{ip} \mid 1 \leq i \leq n, 0 \leq p \leq 5\}$ son dados como input, indicando claramente las estructuras de datos utilizadas y la complejidad resultante. Nota: para que el ejercicio se considere bien es necesario que la complejidad temporal del algoritmo sea $O(n^2)$.
 - d) Escribir el (pseudo-)código del algoritmo top-down resultante.
- 2) Dado un digrafo G y dos vértices s y t , queremos encontrar un recorrido de s a t que tenga longitud par y use la menor cantidad de aristas. Modelar este problema como una instancia particular de camino mínimo que pueda ser resuelto con BFS (sin necesidad de modificar el algoritmo visto en clase; observar que el recorrido podría no existir). **Justificar** que el modelo propuesto resuelve el problema.
 - 3) Decimos que un grafo es *fácil* cuando cada una de sus aristas pertenece a lo sumo a un ciclo del grafo. Diseñar un algoritmo para encontrar un árbol generador mínimo de un grafo fácil G en $O(|V(G)|)$ tiempo. El algoritmo debe ser *robusto* en el sentido de que debe reportar un error en caso que G no sea fácil. **Justificar** que el algoritmo propuesto es correcto. **Ayuda:** para el algoritmo, separe su algoritmo en dos partes; primero encuentre un árbol generador (apropiado) de G en tiempo lineal; luego “corrija” el árbol para obtener uno mínimo. Para la complejidad, determine cuántas aristas puede tener un grafo fácil.
 - 4) Diseñar un algoritmo eficiente que dado un digrafo G con pesos no negativos, dos vértices s y t y una cota c , determine una arista de peso máximo de entre aquellas que se encuentran en algún camino de s a t cuyo peso (del camino, no de la arista) sea a lo sumo c . **Justificar** que el algoritmo propuesto es correcto.